เขมะทัต วิภาตะวนิช

สื่อสารข้อมูลโดยใช้ชุดโปรโตคอล TCP/IP (ตอนจบ)

กล่าวถึงรายละเอียดของชุดโปรโตคอล TCP/IP และการทำงานร่วมกันของโปรโตคอลแต่ละชั้น

ในตอนต่อของบทความนี้เป็นการกล่าวถึงรายละเอียดของโปรโตคอล IP, TCP และ UDP รวมทั้งการทำงานร่วมกันของโปรแกรมประยุกต์กับชุดโปรโตคอล TCP/IP และโปรโตคอลในชั้นที่สอง เพื่อแสดงให้เห็นว่าการทำงานครั้งหนึ่ง ๆ นั้นมีเหตุการณ์อะไรเกิดขึ้นบ้าง

โปรโตคอล IP

โปรโตคอล IP เป็นโปรโตคอลที่ให้บริการแก่โปรโตคอลชั้นที่สี่ (TCP, UDP) ในแบบที่เรียกว่า "connectionless :CL" นั่นก็คือ ไม่มีการกำหนดหรือหาเส้นทางระหว่างต้นทางกับปลายทางเอาไว้ก่อน แต่การหาเส้นทางขึ้นอยู่กับอุปกรณ์หาเส้นทาง (router) ซึ่งก็หมายความว่าภายใน IP แพคเก็ตจะต้องมีสารสนเทศที่สำคัญและเพียงพอต่ออุปกรณ์ดังกล่าวในการนำส่ง IP แพคเก็ตไปยังปลายทาง ได้รูปแบบของโปรโตคอล IP แสดงในรูปที่ 1 มีรายละเอียดดังนี้

Version (ver) มีขนาด 4 บิต แสดงถึงรุ่นของโปรโตคอล IP เนื่องจากการตีความในเขตต่าง ๆ ของโปรโตคอล IP ใช้ซอฟต์แวร์เป็นตัวจัดการ ดังนั้นจึงจำเป็นต้องมีการบ่งบอกถึงรุ่นของโปรโตคอล IP ไว้ด้วย ในกรณีที่รุ่นไม่ตรงกันจะไม่มีการตีความเกิดขึ้น IP แพคเก็ตที่ใช้งานอยู่ในขณะนี้เป็นรุ่นที่ 4 (ver=4)

Header (head) มีขนาด 4 บิต บอกถึงความยาวเฉพาะส่วนหัวของ IP แพคเก็ต (ส่วนหัวของ IP แพคเก็ต เริ่มนับจากเขต version ไปจนถึงไบต์สุดท้ายก่อนที่จะถึงเขต data) หน่วยนับเป็นจำนวนเท่าของ 4 ไบต์ เช่น ถ้า header มีค่าเท่ากับ 5 หมายถึงส่วนหัวมีขนาด 20 ไบต์ เป็นต้น

Type of Service (TOS) มีขนาด 8 บิต แบ่งออกเป็น 6 เขตย่อยดังแสดงในรูปที่ 2

รูปที่ 1
เขตต่าง ๆ ของ IP แพคเก็ต

ver	head	TOS	total length
identification		D M	frag. offset
TTL	protocol		header checksum
source IP address			
destination IP address			
options			padding
data			

รูปที่ 2
เขตย่อยต่าง ๆ ภายในเขต TOS

precedence	D	T	R	C	unused

- precedence มีขนาด 3 บิต ใช้ในการกำหนดลำดับความสำคัญของ IP แพคเก็ต ในกรณีที่ปริมาณของแพคเก็ตมีอยู่มากในระบบ อุปกรณ์หาเส้นทางจะไม่ทำการหาเส้นทางให้กับแพคเก็ตที่มีลำดับความสำคัญต่ำสุด หรือกล่าวง่าย ๆ ก็คือตัดแพคเก็ตนั้นออกจากระบบ

- D บิต มีขนาด 1 บิต ใช้บอกถึงความต้องการการให้บริการจากเครือข่าย ถ้าบิตนี้มีค่าเป็น "1" หมายถึงต้องการเส้นทางที่มีการหน่วงเวลา (delay time) ต่ำที่สุดเท่าที่ระบบจะหาให้ได้

- T บิต มีขนาด 1 บิต ถ้าบิตนี้มีค่าเป็น "1" หมายถึงต้องการเส้นทางที่มีความสามารถส่งผ่านข้อมูลได้ปริมาณมาก ๆ ในหนึ่งหน่วยเวลา

- R บิต มีขนาด 1 บิต ถ้าบิตนี้มีค่าเป็น "1" หมายถึงต้องการเส้นทางหรือบริการที่มีความเชื่อถือได้สูง

- C บิต มีขนาด 1 บิต ถ้าบิตนี้มีค่าเป็น "1" หมายถึงต้องการเส้นทางที่มีค่าพารามิเตอร์ของ "cost" ต่ำ (ในส่วนของการหาเส้นทางนั้น cost ต่ำหมายถึงเป็นเส้นทางที่ดี)

- unused มีขนาด 1 บิต ยังไม่มีการนำบิตนี้มาใช้งาน

Total length มีขนาด 16 บิต ใช้บอกถึงความยาวของ IP แพคเก็ตทั้งหมด (ส่วนหัวและส่วนข้อมูล) หน่วยนับเป็นจำนวนเท่าของไบต์ ดังนั้น IP แพคเก็ตมีความยาวสูงสุดเท่ากับ 2^{16} หรือ 65,536 ไบต์

Identification มีขนาด 16 บิต ใช้บอกถึงหมายเลขของ IP แพคเก็ต เขตนี้ถูกกำหนดขึ้นมาเพื่อไม่ให้สถานีปลายทางสับสนว่าตัวมันเองได้รับแพคเก็ตที่สมควรได้รับแล้วหรือไม่ ท่านผู้อ่านต้องคิดไว้เสมอว่าการส่งผ่านข้อมูลในลักษณะ connectionless นั้นมีโอกาสที่ข้อมูลจะสูญหายหรือซ้ำได้ ดังนั้นการกำหนดหมายเลขนี้ขึ้นมาก็เพื่อป้องกันการสับสนที่อาจจะเกิดขึ้นได้

D บิต มีขนาด 1 บิต (อย่าสับสนกับ "D บิต" ที่อยู่ในเขตย่อย TOS นะครับ) ถ้าบิตนี้มีค่าเป็น "1" หมายความว่าห้ามตัดทอน IP แพคเก็ตออกเป็นแพคเก็ตย่อยโดยเด็ดขาด แต่ถ้าบิตนี้มีค่าเป็น "0" หมายความว่าสามารถตัดทอน IP แพคเก็ตออกเป็นแพคเก็ตย่อยได้ การกระทำเช่นนี้มีข้อดีคือ เมื่อ IP แพคเก็ตผ่านจากระบบหนึ่งไปสู่ระบบอื่น ๆ ที่มีขนาดของแพคเก็ตที่ต่างกัน อุปกรณ์หาเส้นทางสามารถแบ่งแพคเก็ตขนาดใหญ่นั้นให้มีขนาดเล็กลงและพอดีกับระบบใหม่ได้ ทำให้มีความยืดหยุ่นและได้ประโยชน์สูงสุดในการส่งข้อมูลผ่านระบบเครือข่ายที่มีขนาดแพคเก็ตไม่เท่ากัน ซึ่งลักษณะเช่นนี้เป็นเรื่องสามัญในโลกของเน็ตเวอร์ก ดังแสดงในรูปที่ 3ก.

ถ้าท่านผู้อ่านยังจำได้ สำหรับโปรโตคอลอีเธอร์เนตและโทเก้นริงนั้นไม่มีส่วนใดเลยที่อนุญาตให้กระทำเช่นนี้ได้ (ถ้าจำไม่ได้ให้ลองกลับไปอ่านไมโครฯ ฉบับที่ 113 ดูนะครับ) ท่านผู้อ่านลองนึกภาพง่าย ๆ ดูว่า ถ้าต้องการเชื่อมระบบอีเธอร์เนตซึ่งมีขนาดความยาวของข้อมูลสูงสุดเป็น 1,500 ไบต์เข้ากับระบบโทเก้นริงที่มีขนาดความยาวของข้อมูล 4,096 ไบต์ จะเห็นได้ว่าการส่งข้อมูลจากอีเธอร์เนตไปยังโทเก้นริงสามารถทำได้โดยตรง แต่ในทางตรงกันข้าม จากโทเก้นริงไปยังอีเธอร์เนตไม่สามารถทำได้ แต่ถ้าต้องการก็จำเป็นต้องลดขนาดข้อมูลของโทเก้นริงให้เป็น 1,500 ไบต์เสียก่อน ซึ่งหมายความว่าทุก ๆ สถานีภายในโทเก้นริงต้องใช้ความยาวข้อมูลขนาด 1,500 ไบต์เหมือนกันหมด ถึงแม้ว่าจะเป็นการส่งระหว่างสถานีภายในระบบโทเก้นริงเองก็ตาม

อนึ่ง การแบ่งแพคเก็ต IP ออกเป็นแพคเก็ตย่อยนั้น แต่ละแพคเก็ตย่อยต้องมีหมายเลข IP แพคเก็ตเดียวกันกับแพคเก็ตใหญ่เสมอ

M บิต มีขนาด 1 บิต ถ้าบิตนี้ถูกกำหนดเป็น "1"ในแพคเก็ตใด หมายความว่าเป็นแพคเก็ตย่อยที่ถูกแบ่งออกมาจากแพคเก็ตใหญ่โดยมีหมายเลข IP

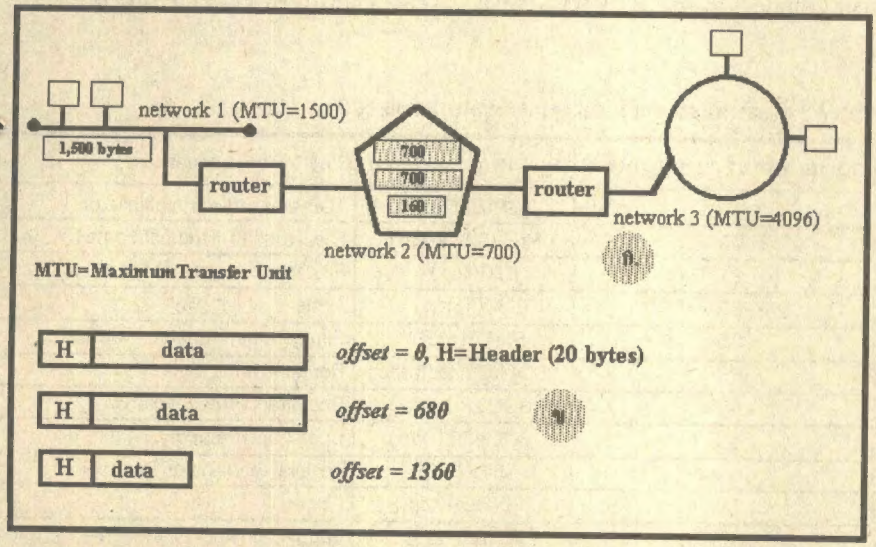
แพคเก็ตเหมือนกับแพคเก็ตใหญ่ เพื่อที่ปลายทางจะสามารถรวบรวมแพคเก็ตย่อยเหล่านี้ให้กลับคืนมาเป็นแพคเก็ตใหญ่ได้ตามเดิม แต่ถ้าบิตนี้เป็น "0" หมายความว่าเป็นแพคเก็ตเดียวโดด ๆ หรืออาจจะเป็นแพคเก็ตย่อยตัวสุดท้ายของแพคเก็ตใหญ่ก็ได้ ซึ่งรายละเอียดจะต้องพิจารณาจากเขต fragment offset ประกอบด้วย อนึ่ง ตามข้อกำหนดของโปรโตคอล IP ถึงแม้ว่าอุปกรณ์หาเส้นทางสามารถแบ่งแพคเก็ตใหญ่ออกเป็นส่วนย่อยได้ก็ตาม แต่ไม่สามารถรวบรวมกลับคืนมาได้ ซึ่งหน้าที่นี้เป็นของสถานีปลายทางเท่านั้น

Fragment offset (frag. offset) มีขนาด 13 บิต ใช้ในการพิจารณาร่วมกับเขต identification, total length, D บิต และ M บิต เพื่อเรียงลำดับและหาขนาดความยาวของแพคเก็ตที่แท้จริงที่ถูกส่งออกมาจากสถานีต้นทาง ดังแสดงในรูปที่ 3ข. ขณะที่มีการแบ่งแพคเก็ตเกิดขึ้น ค่าออฟเซตจะเปลี่ยนไป ดังนั้นเมื่อนำแพคเก็ตขนาดเล็กที่มีหมายเลขของ IP เดียวกันกับที่รับได้มาเรียงตามค่าของออฟเซตที่ถูกต้องโดยดู M บิตประกอบก็จะได้แพคเก็ตเดิมกลับคืนมา

Time to live (TTL) มีขนาด 8 บิต เป็นเขตที่ใช้ในการกำหนดอายุของ IP แพคเก็ต โดยปกติอายุของแพคเก็ตเท่ากับ 255 (2^8-1) ทุกครั้งที่แพคเก็ตผ่านอุปกรณ์หาเส้นทาง ค่าของ TTL จะลดลงหนึ่งเสมอ ถ้าค่าของ TTL เป็นศูนย์ก็หมายความว่าหมดอายุขัย เมื่ออุปกรณ์หาเส้นทางตัวใดพบสภาพเช่นนี้ก็จะนำแพคเก็ตนั้น ๆ ออกจากระบบทันที การกำหนดให้มีเขตนี้ก็เพื่อป้องกันการเกิดแพคเก็ตวนอยู่ในระบบตลอดกาล ซึ่งก่อให้เกิดความเสียหายแก่ระบบโดยรวม เช่น เป็นสาเหตุให้มีปริมาณแพคเก็ตในระบบมาก สถานีปลายทางอาจจะรับแพคเก็ตผิดได้ เป็นต้น

Protocol มีขนาด 8 บิต เป็นเขตที่บอกชนิดโปรโตคอลที่ถูกห่อหุ้ม (encapsulate) โดยส่วนหัวของ IP แพคเก็ต ซึ่งอาจจะเป็นโปรโตคอลชั้นที่สามหรือสี่ก็ได้ ดังแสดงในตารางที่ 1

Header checksum มีขนาด 16 บิต เป็นเขตที่ใช้ในการตรวจสอบความ



รูปที่ 3ก. แพคเก็ตขนาด 1,500 ไบต์ถูกแบ่งย่อยเป็น 3 แพคเก็ตเมื่อผ่านเข้าสู่เครือข่ายที่ 2
3ข. ค่า fragment offset ที่บ่งถึงลำดับของแพคเก็ต

ตารางที่ 1 ตัวอย่างของหมายเลขที่แสดงถึงชนิดของโปรโตคอล

protocol	ชนิดของโปรโตคอล
1	ICMP
6	TCP
16	CHAOS
17	UDP
89	OSPF

ผิดพลาดเฉพาะส่วนหัวของ IP แพคเก็ต การคำนวณจะใช้วิธีการบวกเลข 16 บิต แบบ 1's complement เมื่อได้ผลลัพธ์แล้วก็ทำการผกผัน (inverse) ผลลัพธ์อีกครั้งหนึ่งจึงจะได้ checksum ที่แท้จริง การตรวจสอบกระทำเฉพาะส่วนหัวของข้อมูล เพราะส่วนข้อมูลของ IP แพคเก็ตจะเป็นโปรโตคอลชั้นที่สี่ซึ่งมี checksum เป็นของตัวเองอยู่แล้ว อนึ่ง checksum ของ IP แพคเก็ตอาจจะไม่ถูกนำมาใช้งานก็ได้ถ้า IP แพคเก็ตถูกห่อหุ้มโดยโปรโตคอลชั้นที่สองที่ใช้ checksum ที่มีความเชื่อถือได้ เช่น อีเธอร์เนต ในกรณีที่ checksum ของ IP แพคเก็ตไม่ถูกใช้งานค่าของมันจะเป็นศูนย์ขนาด 16 บิต

Source/Destination address มีขนาดอย่างละ 32 บิต ซึ่งก็คือ IP แอดเดรสดังที่กล่าวไปแล้วในตอนที่ 1

Option ส่วนของ option นี้อาจจะมีหรือไม่มี หรือมีขนาดเท่าใดขึ้นอยู่กับชนิดของ IP แพคเก็ต เนื่องจากรายละเอียดของ option ค่อนข้างปลีกย่อย ผู้เขียนจึงไม่ขอกล่าวในที่นี้

Padding มีขนาด 0-3 ไบต์ ใช้เป็นส่วนที่ทำให้ขนาดของ option เป็นจำนวนเท่าของ 32 บิต

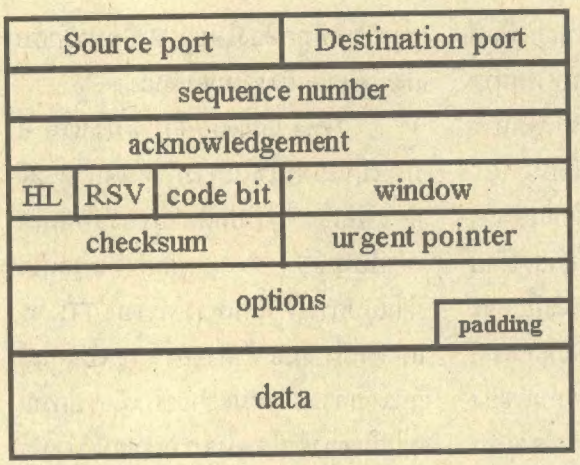
Data เป็นส่วนของข้อมูลหรือโปรโตคอลที่อยู่ในชั้นที่สูงกว่าหรือเท่ากับ IP แพคเก็ต

โปรโตคอล TCP

โปรโตคอล TCP เป็นโปรโตคอลที่อยู่ในชั้นที่สี่ มีการให้บริการในแบบ "connection-oriented: CO" ซึ่งหมายความว่าก่อนที่จะมีการรับส่งข้อมูลระหว่างต้นทางกับปลายทางจะต้องมีการสถาปนาการเชื่อมต่อขึ้นมาเสียก่อน และเมื่อสิ้นสุดการส่งผ่านข้อมูลก็จะยกเลิกการสถาปนานั้น ๆ ท่านผู้อ่านคงจะสงสัยว่าทำไมต้องทำให้บริการของ TCP แตกต่างไปจาก IP ผู้เขียนใคร่ขอสร้างความกระจ่างที่ประเด็นนี้ก่อนแล้วจึงค่อยอธิบายเขตต่าง ๆ ของรูปแบบโปรโตคอล TCP

IP เป็น connectionless เพราะคณะผู้ออกแบบชุดโปรโตคอล TCP/IP เชื่อมั่นว่าระบบเครือข่ายที่ให้บริการในการส่งผ่านข้อมูลย่อมจะมีการผิดพลาดเกิดขึ้นได้เสมอ (อย่าลืมนะครับว่าชุดโปรโตคอลนี้ถูกออกแบบในทศวรรษที่ 70 ซึ่งในขณะนั้นการสื่อสารข้อมูลยังมี Bit Error Rate (BER) สูง คือประมาณ 10^{-3}-10^{-4}) และการทำให้แต่ละโหนดสถาปนาวงจรขึ้นมาทุกครั้งที่มีการรับส่งข้อมูลจากต้นทางไปยังปลายทางทำให้เกิดความยุ่งยากอย่างมากในระบบเครือข่าย

ด้วยสองสาเหตุหลักนี้ทำให้การออกแบบโปรโตคอลเลือกที่จะใช้ CL สำหรับ IP แพคเก็ต เมื่อโปรโตคอลในระดับล่างให้บริการในลักษณะเช่นนี้ ผู้ออกแบบต้องยอมรับสภาพการมีแพคเก็ตผิดพลาด แพคเก็ตซ้ำหรือแพคเก็ตสูญหาย เพื่อแก้ไขปัญหาเช่นนี้ก็ไม่มีอะไรดีไปกว่าการเลือกใช้ CO สำหรับโปรโตคอลในชั้นที่อยู่เหนือกว่า ซึ่งโปรโตคอล TCP ก็เลือกใช้วิธีนี้ แต่อย่าลืมนะครับว่าคู่แฝดของ TCP คือ UDP นั้นมีการให้บริการในลักษณะ CL ซึ่งรายละเอียดจะขอกล่าวในภายหลัง คำศัพท์ที่ใช้เรียก TCP แพคเก็ตคือเซกเมนต์ (segment) ดังนั้นเมื่อกล่าวถึง TCP และ UDP แพคเก็ตจะใช้คำว่าเซกเมนต์แทน สำหรับรายละเอียดเขตต่าง ๆ ของเซกเมนต์ TCP แสดงในรูปที่ 4 ซึ่งอธิบายได้ดังนี้



รูปที่ 4
เขตต่าง ๆ ของ TCP เซกเมนต์

Source port มีขนาด 16บิต การเข้าถึงแอพพลิเคชันของชุดโปรโตคอล TCP/IP ใช้หมายเลขพอร์ตเป็นตัวกำหนด ดังนั้น source port หมายถึงหมายเลขพอร์ตของสถานีต้นทาง

Destination port มีขนาด 16บิต เป็นหมายเลขพอร์ตของสถานีปลายทางที่ต้องการจะติดต่อด้วย แอพพลิเคชันมาตรฐานจะมีหมายเลขพอร์ตคงที่ เช่น Telnet ใช้พอร์ตหมายเลข 23, FTP (File Transfer Protocol) ใช้พอร์ตหมายเลข 20, 21 เป็นต้น ตารางที่ 2 แสดงถึงบางส่วนของหมายเลขพอร์ตที่มีการใช้งาน สำหรับพอร์ตมาตรฐานถูกกำหนดไว้ที่หมายเลข 0-255 ในส่วนที่นอกเหนือจากนี้ผู้เขียนแอพพลิเคชันสามารถกำหนดเองได้

Sequence number มีขนาด 32 บิต TCP เซกเมนต์ใช้หมายเลขลำดับนี้ในการบอกถึงตำแหน่งไบต์แรกของข้อมูลที่ปรากฏอยู่ในส่วนของเขต data หมายความว่าถ้าข้อมูลและส่วนหัวของ TCP เซกเมนต์ที่จะส่งมีความยาวเกินกว่า

ตารางที่ 2 ตัวอย่างหมายเลขพอร์ตมาตรฐานของชุดโปรโตคอล TCP/IP

หมายเลขพอร์ต (เลขฐานสิบ)	คำสำคัญ (keyword)	หมายเหตุ (comment)
1	TCPMUX(tcp)	TCP port service multiplexor
20	FTP-DATA(tcp)	File Transfer Protocol (data)
21	FTP(tcp)	File Transfer Protocol (control)
23	TELNET(tcp)	Telnet
25	SMTP(tcp)	Simple Mail Transfer Protocol
53	DOMAIN(tcp&udp)	Domain Name Server
67	BOOTPS(udp)	Bootstrap Protocol Server
68	BOOTPC(udp)	Bootstrap Protocol Client
69	TFTP(udp)	Trivial File Transfer Protocol
79	FINGER(tcp)	Finger
161	SNMP(udp)	Simple Network Management Protocol
162	SNMPTRAP(udp)	SNMPTRAP
194	IRC(tcp)	Internet Relay Chat protocol

แพคเก็ตของ IP ที่จะยอมรับได้ (65,536 ไบต์) จึงจำเป็นต้องแบ่งข้อมูลที่จะส่งออกเป็น TCP เซกเมนต์ย่อย ๆ หลายเซกเมนต์ และเนื่องจากการมองข้อมูลของ TCP เซกเมนต์มองเป็น "stream" คือการมองข้อมูลแบบต่อเนื่องกันไปเปรียบเสมือนการนับเลขจำนวน เมื่อข้อมูลในเซกเมนต์แรกถูกตัดตอนส่งที่ไบต์สุดท้ายตำแหน่งใด การเริ่มต้นของเซกเมนต์ใหม่ก็จะเริ่มในตำแหน่งของไบต์ที่อยู่ถัดมาเป็นเช่นนี้เรื่อย ๆ ไปจนสิ้นสุดข้อมูลทั้งหมด จะเห็นได้ว่าเลขลำดับมีลำดับที่ไม่ซ้ำกันฉันใด ลำดับของข้อมูลที่อยู่เรียงกันก็ย่อมไม่ซ้ำกันฉันนั้น ถ้าท่านผู้อ่านนำแนวความคิดการส่งข้อมูลแบบ stream ของ TCP ไปเปรียบเทียบกับการส่งข้อมูลของโปรโตคอลชั้นที่สอง เช่น HDLC (High-level Data Link Control) ซึ่งใช้หมายเลขลำดับเป็นตัวบอกถึงหมายเลขของเฟรมในการรับส่งซึ่งแตกต่างจากแนวความคิดแบบ stream อย่างสิ้นเชิง

Acknowledgement number มีขนาด 32 บิต โปรโตคอล TCP ใช้หลักการของ positive ack. ในการตอบกลับไปยังต้นทางเพื่อแสดงว่าปลายทางได้รับข้อมูลเรียบร้อยแล้ว โดยหมายเลขของ ack. no. จะต้องมีค่าเท่ากับ "ไบต์สุดท้ายบวกหนึ่ง" ของลำดับข้อมูลที่รับได้ (เช่น ไบต์สุดท้ายที่รับได้มีค่าเท่ากับ 1,024 เพราะฉะนั้นหมายเลขของ ack. no. จะต้องมีค่าเท่ากับ 1,025 (1,024+1)) ซึ่งเป็นการบอกต้นทางอย่างเป็นนัยว่า "รับไบต์สุดท้ายได้แล้ว คาดว่าไบต์เริ่มต้นต่อไปที่จะได้รับคือไบต์สุดท้ายบวกหนึ่ง"

Header length (HL) มีขนาด 4 บิต เป็นการบอกถึงความยาวส่วนหัวของ TCP เซกเมนต์ มีหน่วยเป็นจำนวนเท่าของ 4 ไบต์เช่นเดียวกับ IP แพคเก็ต หรืออีกนัยหนึ่งก็คือเป็นการบอกถึงตำแหน่งเริ่มต้นของไบต์ข้อมูล เพราะส่วนหัวของ TCP เซกเมนต์มีความยาวที่เปลี่ยนแปลงได้ขึ้นอยู่กับเขต option โดยทั่วไปถ้าไม่มีในส่วนของ option ค่าของ HL จะเท่ากับ 5 (5x4=20ไบต์)

Reserved (RSV) มีขนาด 4 บิต เป็นส่วนที่ถูกสำรองไว้ใช้ในอนาคต

Code bit ประกอบด้วย 6 เขตย่อย ซึ่งแต่ละเขตย่อยมีขนาด 1 บิต และมีรายละเอียดดังนี้

- เขตย่อย URGent ถ้าบิตนี้เป็น "1" หมายความว่าเขต Urgent pointer (จะกล่าวในภายหลัง) บรรจุข่าวสารที่มีความหมายจะต้องนำมาดีความด้วย การมองข้อมูลของ TCP เซกเมนต์ว่ามีความเร่งด่วน (urgent) นั่นหมายความว่าข้อมูลนั้นไม่ใช่ข้อมูลธรรมดา แต่มีความสำคัญต่อระบบทั้งหมด ดังนั้นเมื่อปลายทางรับข้อมูลประเภทนี้ได้ต้องรีบให้ความสนใจทันที

- เขตย่อย ACKnowledgement ถ้าบิตนี้เป็น "1" หมายความว่าเขต Acknowledgement number มีความหมายต้องนำมาพิจารณาในการตรวจสอบลำดับในการรับส่งข้อมูลด้วย

- เขตย่อย PUSH ถ้าบิตนี้เป็น "1" หมายความว่าทันทีที่สถานีปลายทางรับ TCP เซกเมนต์ได้ต้องรีบส่งข้อมูลนี้ไปยังแอพพลิเคชันทันทีโดยไม่ต้องรอให้บัฟเฟอร์ในการรับข้อมูลเต็มเสียก่อน โดยทั่วไปแล้วการส่งข้อมูลจาก TCP ขึ้นไปยังแอพพลิเคชันต้องรอให้ข้อมูลเข้ามาอยู่ในบัฟเฟอร์เต็มเสียก่อนเพื่อลดจำนวน overhead ที่จะเกิดขึ้นในการส่งโปรโตคอลระหว่างชั้น

- เขตย่อย RESET ถ้าบิตนี้เป็น "1" หมายความว่ามีการผิดพลาดเกิดขึ้นระหว่างคู่สถานีที่มีการเชื่อมต่อถึงกันอยู่และจำเป็นต้องยกเลิกการเชื่อมต่อนี้ แต่ถ้าต้องการส่งข้อมูลถึงกันอีกก็ต้องสถาปนาการเชื่อมต่อขึ้นมาใหม่

- เขตย่อย SYNchronize ถ้าบิตนี้เป็น "1" หมายความว่าขอเริ่มต้นสถาปนาการเชื่อมต่อ และเมื่อการสถาปนาเสร็จสิ้น บิตนี้จะถูกกำหนดให้เป็น "0" และหลังจากนั้นก็สามารถส่งผ่านข้อมูลระหว่างสถานีได้

- เขตย่อย FIN ถ้าบิตนี้เป็น "1" หมายความว่าขอยกเลิกการเชื่อมต่อระหว่างสถานีเพราะไม่มีข้อมูลที่จะส่งถึงกันอีกแล้ว

Window มีขนาด 16 บิต สถานีปลายทางใช้เขตนี้ในการบ่งบอกถึงขนาดของบัฟเฟอร์ (จำนวนไบต์) ที่สามารถรับข้อมูลจากต้นทางได้ หรือเป็นการบอกอย่างเป็นนัยว่า "ถ้าต้นทางส่งปริมาณข้อมูลมามากกว่าที่ปลายทางจะรับได้ อาจมีบางส่วนของข้อมูลสูญหายได้" ข้อดีของเขตนี้ก็คือ เป็นการควบคุมปริมาณข้อมูลในระบบไม่ให้มีมากเกินไป ซึ่งอาจจะทำให้เกิด traffic jam ในระบบ

เครือข่ายได้ ท่านผู้อ่านที่คุ้นเคยกับโปรโตคอลชั้นที่สองเช่น HDLC อาจจะมีคำถามว่า window ของ TCP แตกต่างอย่างไรเมื่อเปรียบเทียบกับ HDLC ความแตกต่างที่เห็นได้ชัดคือ window ของ TCP สามารถเปลี่ยนแปลงขนาดได้ระหว่างการส่งผ่านข้อมูลครั้งหนึ่ง ๆ เพราะที่เวลาหนึ่ง ๆ สถานีปลายทางอาจจะมีขนาดของบัฟเฟอร์ไม่คงที่ แต่ window ของ HDLC จะมีขนาดคงที่ตลอดช่วงการส่งผ่านข้อมูล เพราะขณะที่เริ่มต้นสถาปนาการเชื่อมต่อในระดับโปรโตคอลชั้นที่สองจะมีการตกลงระหว่างสถานีต้นทางและปลายทางว่าต้องการขนาดของ window เป็นเท่าใด ซึ่งมีให้เลือกสองขนาดคือ modulo-8 และ modulo-128

Checksum มีขนาด 16 บิต ใช้วิธีการเช่นเดียวกับ IP แพคเก็ต แต่จะมีส่วนที่นำมาใช้ในการคำนวณเพิ่มเติมนอกเหนือไปจากส่วนหัวของ TCP เซกเมนต์ ซึ่งจะขอกล่าวในภายหลัง

Urgent pointer มีขนาด 16 บิต เปรียบเสมือนเป็นค่า offset ของ sequence number โดยจะบอกถึงตำแหน่งของไบต์สุดท้ายภายใน TCP เซกเมนต์ที่เป็นข้อมูลเร่งด่วน (ข้อมูลภายใน TCP เซกเมนต์หนึ่ง ๆ อาจจะไม่มีความเร่งด่วนทั้งหมดก็ได้) ที่ต้องการให้แอพพลิเคชันพิจารณาทันที อนึ่ง ค่าที่ถูกบรรจุในเขตนี้จะมีความหมายก็ต่อเมื่อเขตย่อย urgent ถูกกำหนดค่าเป็น "1"

Option ส่วนของ option นี้อาจจะมีหรือไม่มี หรือมีขนาดเท่าใดขึ้นอยู่กับชนิดของ TCP เซกเมนต์ เนื่องจากรายละเอียดของ option ค่อนข้างปลีกย่อย ผู้เขียนจึงไม่ขอกล่าวในที่นี้

Padding มีขนาด 0-3 ไบต์ ใช้เป็นส่วนที่ทำให้ขนาดของ option เป็นจำนวนเท่าของ 32 บิต

Source port	Destination port
Length	Checksu
data	

รูปที่ 5
เขตต่าง ๆ ของ UDP เซกเมนต์

Source address		
Destination address		
zero	Protocol	Length

รูปที่ 6
เขตต่าง ๆ ของ pseudo header

โปรโตคอล UDP

โปรโตคอล UDP เป็นคู่แฝดของ TCP แต่มีการให้บริการแบบ connectionless (CL) รูปแบบของโปรโตคอลแสดงในรูปที่ 5 โดยมีรายละเอียดดังนี้

Source & Destination port แสดงในตารางที่ 2 มีความหมายเช่นเดียวกับที่กล่าวในหัวข้อโปรโตคอล TCP ถ้าผู้อ่านสังเกตพอร์ตหมายเลข 53 จะพบว่า ทั้ง TCP และ UDP ใช้พอร์ตหมายเลขเดียวกัน แต่ทำไมสถานีปลายทางจึงแยกแยะได้ว่าเป็น TCP เซกเมนต์หรือ UDP เซกเมนต์ สำหรับคำตอบก็ขอให้ย้อนกลับไปดูที่หัวข้อโปรโตคอล IP ในส่วนของเขต protocol

Length มีขนาด 16 บิต ใช้บอกความยาวของ UDP เซกเมนต์ (นับเป็นจำนวนไบต์)

Checksum มีขนาด 16 บิต มีวิธีการในการคำนวณเช่นเดียวกับ TCP

ในการคำนวณ checksum ของทั้ง TCP และ UDP จะมีการเพิ่ม "pseudo header" เข้ามาในการคำนวณ ดังแสดงในรูปที่ 6 แต่ส่วนที่เพิ่มมานี้ไม่ถูกผนวกเข้าเป็นส่วนหัวหรือข้อมูลของ TCP และ UDP ซึ่งก็หมายความว่าในการส่งผ่าน TCP หรือ UDP เซกเมนต์จะไม่มี pseudo header ปรากฏอยู่เลย pseudo header ประกอบด้วยเขตต่าง ๆ ดังนี้

Source&destination address มีขนาด 32 บิต คือ IP แอดเดรส

Zero มีขนาด 8 บิต เป็นศูนย์หมดทุกบิต

Protocol มีขนาด 8 บิต มีความหมายเช่นเดียวกับที่กล่าวในหัวข้อโปรโตคอล IP

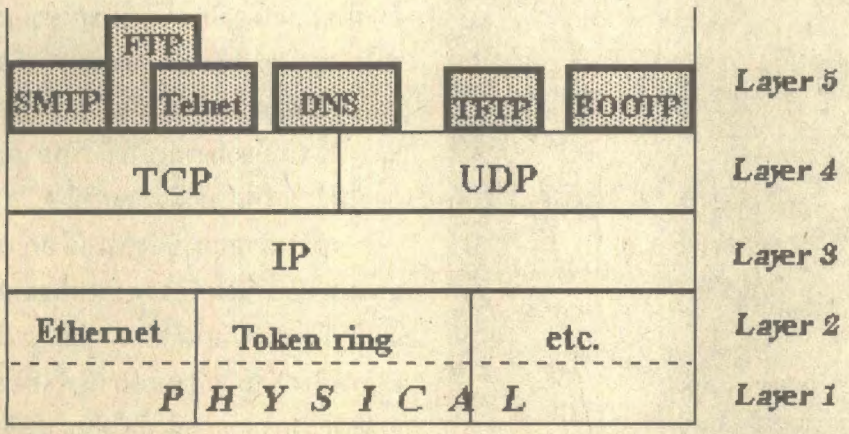
Length มีขนาด 16 บิต เป็นความยาวของ TCP หรือ UDP เซกเมนต์ ซึ่งไม่รวม pseudo header

ท่านผู้อ่านอาจจะสงสัยว่าทำไมต้องมี pseudo header ด้วย การสื่อสารข้อมูลโดยใช้ชุดโปรโตคอล TCP/IP นั้นอาศัยทั้ง IP แอดเดรสและหมายเลขพอร์ต (ซึ่งรวมเรียกว่าหมายเลขซ็อกเก็ต (socket number)) ในการสถาปนาการเชื่อมต่อและแยกแยะแอพพลิเคชันออกจากกันเพื่อป้องกันการสับสนระหว่างสถานีต้นทางและปลายทาง การใช้ pseudo header เป็นส่วนหนึ่งของการคำนวณ checksum ก็เพื่อให้ปลายทางมีความมั่นใจต่อแพคเก็ตหรือเซกเมนต์ที่ได้รับ เพราะใน IP แพคเก็ตและ TCP หรือ UDP เซกเมนต์จะมีส่วนของ IP แอดเดรสและหมายเลขพอร์ตตามลำดับ เมื่อนำทั้งสองส่วนนี้มารวมกันให้มีรูปแบบเหมือน pseudo header ที่ต้นทางและทำการคำนวณโดยวิธีเดียวกัน ผลลัพธ์ย่อมเป็น checksum ตัวเดียวกันถ้า

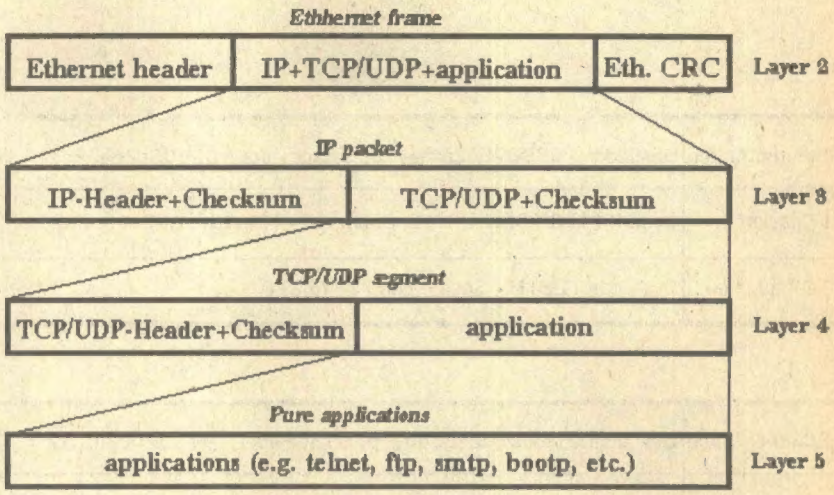
แพคเก็ตและเซกเมนต์ที่ได้รับนั้นถูกต้อง แต่ถ้ามีการผิดพลาดของข้อมูลเกิดขึ้น ค่าที่ได้จากการคำนวณ checksum ย่อมแตกต่างไปจากเดิมด้วย

ความสัมพันธ์และการทำงานร่วมกันของโปรโตคอลแต่ละชั้น

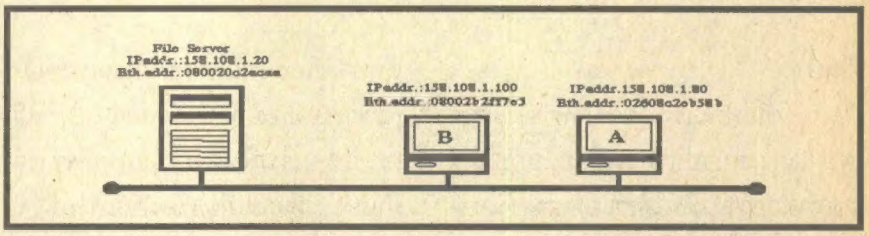
ตัวอย่างลำดับชั้นของชุดโปรโตคอล TCP/IP และแอพพลิเคชัน แสดงในรูปที่ 7 จากรูปจะเห็นได้ว่าแอพพลิเคชันแต่ละประเภทเลือกใช้โปรโตคอลชั้นที่ 4 แตกต่างกัน เช่น SMTP เลือกใช้ TCP ในขณะที่ BOOTP เลือกใช้ UDP อย่างไรก็ตาม มีแอพพลิเคชันบางประเภทเลือกใช้ทั้ง TCP และ UDP สำหรับ TCP และ UDP เลือกใช้โปรโตคอล IP และถัดจากโปรโตคอลชั้นที่ 3 ลงมาก็ขึ้นอยู่กับว่าผู้ใช้งานเลือกใช้เครือข่ายประเภทใดในการส่งผ่านชุดโปรโตคอล TCP/IP ถ้าเป็น LAN ก็อาจจะเลือกใช้อีเธอร์เนตหรือ LAN แบบอื่นที่เห็นว่าสมควร แต่ถ้าเป็น WAN ก็อาจจะใช้ PPP (Point-to-Point Protocol) หรือ HDLC เป็นต้น สำหรับชั้น physical นั้นขึ้นอยู่กับว่าเลือกใช้ LAN แบบใด เพราะ LAN แต่ละแบบจะมีอุปกรณ์เชื่อมต่อต่างกัน สำหรับ WAN นั้นมีให้เลือกหลายมาตรฐาน เช่น RS-232C, CCITT V.35 เป็นต้น



รูปที่ 7 ลำดับชั้นของแอพพลิเคชันและชุดโปรโตคอล TCP/IP



รูปที่ 8 โปรโตคอลในชั้นบนสุดจะถูกห่อหุ้มด้วยโปรโตคอลในชั้นที่ต่ำกว่า



รูปที่ 9 สถานี A และ B ถูกกำหนดให้อยู่ในเน็ตเวอร์กเดียวกันกับไฟล์เซอร์ฟเวอร์

เพื่อให้ง่ายต่อการเข้าใจ ผู้เขียนขอยกตัวอย่างโดยใช้ LAN แบบอีเธอร์เนตประกอบการอธิบาย อย่างไรก็ตาม ท่านผู้อ่านสามารถนำแนวความคิดนี้ไปประยุกต์กับ LAN แบบต่าง ๆ ได้ ถ้าเราลองเขียนความสัมพันธ์ของโปรโตคอลชั้นต่าง ๆ จะได้ดังรูปที่ 8 จะเห็นได้ว่าโปรโตคอลชั้นที่อยู่บนสุดถูกห่อหุ้มด้วยชั้นที่อยู่ถัดมาเป็นเช่นนี้เรื่อย ๆ ไปจนถึงชั้นที่สองซึ่งเป็นการสิ้นสุดการห่อหุ้ม ส่วนชั้น physical นั้นก็ขึ้นอยู่กับว่าสารสนเทศจะถูกส่งออกไปในรูปแบบของสัญญาณไฟฟ้าอย่างไร

จากรูปที่ 9 เราจะมาดูกันว่าถ้าสถานี A ต้องการต่อเข้ากับไฟล์เซอร์ฟเวอร์ (file server) ในลักษณะของเทอร์มินัลระยะไกล (remote terminal) โดยใช้แอพพลิเคชัน "tlet 158.108.1.20" นั้นมีเหตุการณ์คร่าว ๆ อย่างไรเกิด

Ethernet frame	Dest.addr. (080020c2acaa)16, Source addr. (02608c2eb58b)16, Frame type (0800)16
IP packet	Dest.IP (158.108.1.20)10, Source IP (158.108.1.80)10, Protocol (6)10
TCP segment	Dest. port (23)10, Source port (24722)10

ตารางที่ 3
หมายเลขสถานีและพอร์ตที่ถูกบรรจุในเฟรมที่ถูกส่งจากสถานี A ไปยังไฟล์เซอร์ฟเวอร์

Ethernet frame	Dest.addr. (02608c2eb58b)16, Source addr. (080020c2acaa)16, Frame type (0800)16
IP packet	Dest.IP (158.108.1.80)10, Source IP (158.108.1.20)10, Protocol (6)10
TCP segment	Dest. port (24722)10, Source port (23)10

ตารางที่ 4
หมายเลขสถานีและพอร์ตที่ถูกบรรจุในเฟรมที่ถูกส่งจากไฟล์เซอร์ฟเวอร์ไปยังสถานี A

ขึ้นบ้าง

ขั้นตอนแรก (ตารางที่ 3) สถานี A ต้องการสถาปนาการเชื่อมต่อกับไฟล์เซอร์ฟเวอร์ จะเห็นได้ว่าแอดเดรสต้นทาง (ทั้งอีเธอร์เนตและ IP แอดเดรส) เป็นของสถานี A ส่วนแอดเดรสปลายทางเป็นของไฟล์เซอร์ฟเวอร์ สำหรับหมายเลขพอร์ตต้นทางนั้นสถานี A จะทำการสุ่มเลขจำนวนเต็มบวกใด ๆ ขึ้นมา (ห้ามน้อยกว่า 255) ที่ไม่ซ้ำกับหมายเลขพอร์ตที่ปรากฏอยู่บนสถานี A ทั้งหมดในขณะนั้น เพื่อใช้เป็นจุดอ้างอิงของแอพพลิเคชัน แต่สำหรับหมายเลขพอร์ตปลายทางจะเป็นเบอร์ "$(23)_{10}$" เท่านั้น ถ้าเป็นเบอร์อื่น เช่น "$(53)_{10}$" จะกลายเป็น DOMAIN ซึ่งไม่ใช่ TELNET (ดูตารางที่ 2 ประกอบด้วย) เนื่องจากแอพพลิเคชัน "telnet" ใช้บริการของ TCP ดังนั้นหมายเลขโปรโตคอลในส่วนของ IP แพคเก็ตจึงเป็นเบอร์ "$(6)_{10}$" (ดูตารางที่ 1 ประกอบด้วย) และเนื่องจากอีเธอร์เนตให้บริการแก่ IP แพคเก็ต ดังนั้นส่วนของ frame type จึงมีค่าเท่ากับ $(0800)_{16}$ รายละเอียดถูกกล่าวไว้ในบทความตอนที่ 1) จะเห็นได้ว่าอีเธอร์เนตแอดเดรสของทั้งต้นทางและปลายทางเป็นแอดเดรสแบบเฉพาะเจาะจง ถึงแม้ว่าสถานี B รับอีเธอร์เนตเฟรมได้ก็จะไม่สนใจเพราะตัวมันเองไม่มีส่วนเกี่ยวข้องกับเฟรมนี้

ขั้นตอนที่สอง เมื่อไฟล์เซอร์ฟเวอร์ได้รับการขอสถาปนาการเชื่อมต่อจากสถานี A จึงตอบตกลงกลับมายังสถานี A จากตารางที่ 4 จะเห็นได้ว่าแอดเดรสต้นทาง ปลายทาง และหมายเลขพอร์ตมีการสลับกันกับตารางที่ 3 เพราะขณะนี้เรามองสารสนเทศที่ถูกส่งจากไฟล์เซอร์ฟเวอร์ซึ่งเป็นต้นทางไปยังสถานี A ซึ่งเป็นปลายทาง

สถานี A และไฟล์เซอร์ฟเวอร์จะสื่อสารและโต้ตอบกันอีกหลายแพคเก็ต ซึ่งเป็นไปตามแผนภาพสถานะ (state diagram) ของโปรโตคอล TCP/IP ผู้เขียนขออนุญาตไม่นำมากล่าวในที่นี้ เพราะเป็นการยุ่งยากมาก เมื่อสถานี A ได้รับ "prompt" จากไฟล์เซอร์ฟเวอร์ ก็จะใส่ชื่อของผู้ใช้งานลงไป เช่น "login: s34snt" เมื่อไฟล์เซอร์ฟเวอร์ได้รับชื่อแล้วก็ส่งถามรหัสผ่านกลับมายังผู้ใช้งานอีก ผู้ใช้งานก็ต้องใส่รหัสผ่านของตัวเองลงไป เช่น "password: jurassic" เมื่อไฟล์เซอร์ฟเวอร์ตรวจสอบความถูกต้องแล้วผู้ใช้งานก็สามารถเข้าใช้ทรัพยากรของไฟล์เซอร์ฟเวอร์ได้จากสถานี A

จากตัวอย่างข้างต้น เมื่อการสถาปนาการเชื่อมต่อระหว่างสถานี A และไฟล์เซอร์ฟเวอร์เกิดขึ้นโดยสมบูรณ์ แอพพลิเคชันที่อยู่บนสถานีทั้งสองสามารถส่งผ่านข้อมูลถึงกันได้โดยอาศัยหมายเลขพอร์ต (สถานี A ใช้หมายเลข

พอร์ตเบอร์ "24722":A's source port ไฟล์เซอร์ฟเวอร์ใช้หมายเลขพอร์ตเบอร์ "23": file server's source port)

ผู้เขียนใคร่ขอตั้งคำถามถามท่านผู้อ่านสักข้อหนึ่งคือ ขณะที่การเชื่อมต่อระหว่างสถานี A กับไฟล์เซอร์ฟเวอร์ยังคงอยู่นั้น ถ้าสถานี B ต้องการต่อเข้ากับไฟล์เซอร์ฟเวอร์ในลักษณะของเทอร์มินัลระยะไกลเช่นเดียวกับสถานี A จะทำได้หรือไม่? และถ้าทำได้ไฟล์เซอร์ฟเวอร์จะตรวจสอบความแตกต่างระหว่างสถานีทั้งสองได้อย่างไร?

คำถามนี้ต้องการชี้ให้เห็นถึงประเด็นสำคัญประเด็นหนึ่งคือ ทั้งสถานี A และสถานี B ใช้หมายเลขพอร์ตปลายทางเบอร์เดียวกันคือเบอร์ "23" เพื่อต้องการเชื่อมต่อเข้ากับแอพพลิเคชัน "telnet" ถ้าไฟล์เซอร์ฟเวอร์ใช้หมายเลขพอร์ตอย่างเดียวในการจำแนกการเชื่อมต่อจะไม่สามารถทำได้เพราะเป็นหมายเลขเดียวกัน

แต่สำหรับคำตอบของคำถามนี้คือ สามารถทำได้โดยไฟล์เซอร์ฟเวอร์ใช้หมายเลขพอร์ตและ IP แอดเดรส (หมายเลขซ็อกเก็ต) ในการจำแนกว่าเป็นการเชื่อมต่อมาจากสถานีใด เพราะ IP แอดเดรสไม่สามารถซ้ำกันได้เด็ดขาด และนี่ก็คือประโยชน์ของหมายเลขซ็อกเก็ต

บทสรุป

ตัวจักรสำคัญที่ทำให้เครือข่ายโลกาภิวัฒน์หรืออินเตอร์เน็ตเป็นที่ยอมรับกันโดยทั่วไปในยุคปัจจุบันก็คือ ชุดโปรโตคอล TCP/IP ความสัมพันธ์และการทำงานร่วมกันของชุดโปรโตคอล TCP/IP กับแอพพลิเคชัน และโปรโตคอลในระดับล่างที่หยิบยกมาให้ดูนี้ก็เป็นเพียงเสี้ยวหนึ่งของการประยุกต์ใช้งานทั้งหมดที่มีอยู่ ตราบใดก็ตามที่พัฒนาการของมวลมนุษยชาติยังคงมีอยู่อย่างต่อเนื่อง ผู้เขียนก็คงมีเรื่องสนุก ๆ มาเล่าสู่กันฟังอีกนะครับ

เอกสารอ้างอิง

1. Postel J.B., Transmission Control Protocol, RFC793, September 1981.
2. Postel J.B., User Datagram Protocol, RFC768, August 1980.
3. Postel J.B., Internet Protocol, RFC791, September 1981.
4. Postel J.B. and J.K. Raynolds, Telnet Protocol Specification, RFC 854, May 1983.

ตรงนี้

สำหรับผู้ที่ต้องการศึกษาด้วยตนเอง

หนังสือเล่มนี้เขียนขึ้นเพื่อตอบสนองความต้องการของผู้ใช้โปรแกรม Word ที่เพิ่มมากขึ้นตั้งแต่ Word ภาษาไทยออกสู่ท้องตลาด เนื้อหาของหนังสือเริ่มตั้งแต่พื้นฐานการใช้งานวินโดว์จนสามารถใช้งานโปรแกรมได้อย่างสมบูรณ์ เนื่องจากมีเนื้อหาครอบคลุมคำสั่งและเทคนิคการใช้ที่สำคัญ เช่น การจัดรูปแบบตัวอักษรและหน้ากระดาษ การสร้างตาราง การแบ่งคอลัมน์และการนำกราฟิกเข้ามาใช้ในการจัดเอกสาร

หาซื้อได้ตามร้านหนังสือชั้นนำทั่วไป
หรือที่ ซีเอ็ดบุ๊คเซ็นเตอร์ทุกสาขา
และสั่งซื้อได้ที่ซีเอ็ด สำนักงานใหญ่
โทร 248-6280

มีจำหน่ายแล้ว

ซีเอ็ด